Intelligent Data Analysis 17 (2013) 251–270
DOI 10.3233/IDA-130577
IOS Press

A two-phase hybrid of semi-supervised and active learning approach for sequence labeling

Hamed Hassanzadeh[a] and Mohammadreza Keyvanpour[b,*]
[a]*Young Researchers Club, Qazvin Branch, Islamic Azad University, Qazvin, Iran*
[b]*Department of Computer Engineering, Alzahra University, Tehran, Iran*

Abstract. In recent years, many NLP systems and tasks are developed using machine learning methods. In order to achieve the best performance, these systems are generally trained on a large human annotated corpus. Since annotating such corpora is a very expensive and time-consuming procedure, manually annotating corpora is become one of the significant issues in many text based tasks such as text mining, semantic annotation, Named Entity Recognition and generally Information Extraction. Semi-supervised Learning and Active Learning are two distinct approaches that deal with reduction of labeling costs. Based on their natures, Active and semi-supervised learning can produce better results when they are jointly applied. In this paper we propose a combined Semi-Supervised and Active Learning approach for Sequence Labeling which extremely reduces manual annotation cost in a way that only highly uncertain tokens need to be manually labeled and other sequences and subsequences are labeled automatically. The proposed approach reduces manual annotation cost around 90% compare with a supervised learning and 30% in contrast with a similar fully active learning approach. Conditional Random Field (CRF) is chosen as the underlying learning model due to its promising performance in many sequence labeling tasks. In addition we proposed a confidence measure based on the model's variance reduction that reaches a considerable accuracy for finding informative samples.

Keywords: Active learning, semi-supervised learning, sequence labeling, named entity recognition

1. Introduction

Natural Language Processing (NLP) and Information Extraction (IE) with their various tasks and applications such as Part of Speech tagging (POS tagging), Named Entity Recognition (NER), chunking, and semantic annotation, are matters of concern in several fields of computer science for years. NER task is one of the most important subtasks in information extraction [5,8].

Named entity (NE) is structured information referring to predefined proper names like persons, locations and organizations. Automatically extraction of proper names is useful to many problems such as machine translation, information retrieval, question answering and summarization. The goal of Named Entity Recognition is to classify names into some particular categories. It is defined as the identification

*Corresponding author: Mohammadreza Keyvanpour, Department of Computer Engineering, Alzahra University, Tehran, Iran. Tel.: +98 21 8861 7750; Fax: +98 21 8804 1460; E-mail: keyvanpour@alzahra.ac.ir.

of Named Entities (NEs) within text and labeling them by pre-defined classes such as person name, locations, organizations, etc. Each entity is annotated with the type of its expression and its position in the expression, i.e. the beginning or the continuation of the expression [8].

In many works NER task is formulated as a sequence labeling problem and thus can be done with machine learning algorithms supporting sequence labeling task [20,30,32]. It is considerably more difficult to obtain labeled training data for complex structured prediction tasks, such as parsing or sequence modeling (part-of-speech tagging, word segmentation, named entity recognition, and so on), than for classification tasks (such as document classification). This comes from the fact that hand-labeling individual words and word boundaries is much harder than assigning text-level class labels. Moreover, this obstacle may be even more serious with some kind of specific learner models such as Hidden Markov Models (HMM) and Conditional Random Fields (CRFs).

In recent years, more and more NER systems are developed using machine learning methods. In order to achieve the best performance, the systems are generally trained on a large human annotated corpus. However, since annotating such a corpus is very expensive and time-consuming, it is difficult to adapt the existing NER systems to a new application or domain. In order to overcome the difficulty, different machine learning approaches such as Semi-supervised Learning (SSL) and Active Learning (AL) are applied [5,8].

It can be considered that the text data in NER or semantic annotation domains consist of two meaningful parts, i.e. sequences and tokens. By this consideration, in this paper we propose a new hybrid approach by combining AL and SSL in two levels of sequence and token and within two phases. Also we introduced a variance based confidence measure for our combined approach that to the best of our knowledge it is the first attempt to use it in sequence labeling field [5,20].

1.1. Semi-supervised learning

Labeled instances are often difficult, expensive, or time consuming to obtain, as they require the efforts of experienced human annotators. Meanwhile unlabeled data may be relatively easy to collect, but there has been few ways to use them. Semi-supervised learning addresses this problem by using large amount of unlabeled data, together with the labeled data, to build better learners. In other words, semi-supervised learning is halfway between supervised and unsupervised learning and makes use of both labeled and unlabeled data for training. Because semi-supervised learning requires less human effort and gives higher accuracy, it is of great interest both in theory and in practice. There are different semi-supervised learning approaches such as self-training, CO-training, Expectation Maximization (EM), CO-EM, etc. [29].

Self-training is a commonly used technique for semi-supervised learning in which a classifier is first trained with the small amount of labeled data. The classifier is then used to classify the unlabeled data. Typically the most confident unlabeled instances, along with their predicted labels, are added to the training set. The classifier is re-trained and the procedure repeated. Note that the classifier uses its own predictions to teach itself, so it's a highly challenging task to decide which instances must be chosen for self-training. The overall process of self-training is shown in Algorithm 1.

1.2. Active learning

The discipline behind the Active Learning (AL) is that, instead of relying on random sampling from the large training data, it's possible to actively participate in the optimal choosing of training examples which have the greatest impact on the learner model. In AL scenario only examples of high training utility are selected for manual annotation in an iterative manner. The main challenge is to identify "informative"

Algorithm 1 Self-training

```
Given:
Labeled data $\{(x_i, y_i)\}_{i=1}^{l}$, unlabeled data $\{x_i\}_{j=l+1}^{l+u}$
Algorithm:
1. Initially, let $L = \{(x_i, y_i)\}_{i=1}^{l}$ and $U = \{x_i\}_{j=l+1}^{l+u}$.
2. Repeat:
    3. Train a model $(f)$ from $L$ using supervised learning.
    4. Apply $f$ to the unlabeled instances in $U$ to predict possible labels.
    5. Remove a subset $S$ (set of instances which their labels are predicted with confidence) from
    $U$; add $\{(x, f(x))|x \in S\}$ to $L$.
```

instances that should be labeled to improve the model training due to the fact that one could not afford to label all samples [4]. Some of the most common methods of AL are uncertainty sampling [7], query by committee (QBC) [13,31], and query by margin [6].

Different approaches to AL have been successfully applied to a wide range of NLP tasks. A comparison of the effectiveness of these approaches for sequence labeling tasks of NLP is widely discussed in [5]. When used for sequence labeling tasks such as NER, the examples selected by AL are sequences of text, typically sentences. Approaches to AL for sequence labeling are usually unconcerned about the internal structure of the selected sequences [20]. A general sequence-level AL framework for NER is shown in Algorithm 2.

In this paper we present a combined semi-supervised active learning approach for NER task that consider subsequences (tokens) other than sequences while looking for informative instances. By finding the sequences which the current model is certain about labeling them, we use the advantage of unlabeled instances in an additional semi-supervised iteration. We use learner model's variance as a new confidence measure in our proposed framework. For boosting self-training algorithm when it begins from training on a tiny initial labeled data set, we divide unlabeled data set into segments to apply the previous segment's trained model in the next segment.

1.3. Conditional random field for sequence labeling

Similar to the Hidden Markov Models (HMMs) [25], Conditional Random Fields (CRFs) [17] are a probabilistic framework for labeling and segmenting sequential data. A conditional Random fields is an undirected graphical model and calculates the conditional probability of output values based on given input values. To reduce complexity, strong independence assumption is made between observation variables when HMMs is used, which impairs the accuracy of the model. When using CRF, it doesn't need to make assumptions on the dependencies among observation variables, which is different from HMMs.

The CRFs have been applied in many domains to deal with the structured data. Because of its linear structure, Linear-chain CRFs is frequently chosen to deal with the linear labeling questions. Figure 1 shows the graphical representation of liner-chain CRFs.

A linear-chain CRFs model with the observation sequence $\vec{x} = (x_1, \ldots, x_n)$ and the label sequence $\vec{y} = (y_1, \ldots, y_n)$ is described as follow:

$$P(\vec{y}\,|\,\vec{x}) = \frac{1}{Z(\vec{x})}.\exp\left(\sum_{j=1}^{n}\sum_{i=1}^{m}\lambda_i f_i\left(y_{j-1}, y_j, \vec{x}, j\right)\right) \tag{1}$$

In Eq. (1), y_j is the label for position j, state feature function is concerned y_j with the entire observation sequence, the transition feature between labels of position $j-1$ and j on the observation sequence

Algorithm 2 Sequence-level AL framework for NER

```
Given:
$B$: Number of examples to be selected
$L$: Set of labeled examples
$P$: Set of unlabeled examples
$U_M$: Utility function
Algorithm:
Loop until stopping criterion is met
  1. Learn model $M$ from $L$
  2. For all $p_i \in P : u_{p_i} \leftarrow U_M(p_i)$
  3. Select B examples $p_i \in P$ with highest utility $u_{p_i}$
  4. Query human annotator for labels of all $B$ examples
  5. Move newly labeled examples from $P$ to $L$
Return $L$
```

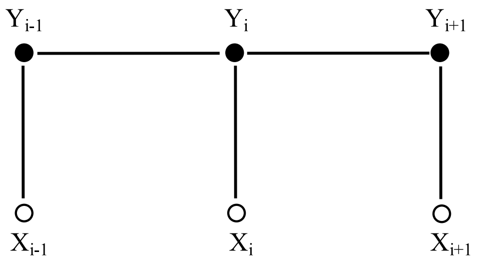

Fig. 1. Graphical representation of linear-chain CRFs [26].

is also considered [14]. Each feature f_i can be either state feature function or transition feature function. λ_s are the parameters to weight corresponding features and can be estimated from training data. Z is a normalization factor defined as:

$$Z = \sum_{\vec{y}} \exp\left(\sum_{j=1}^{n}\sum_{i=1}^{m} \lambda_i f_i\left(y_{j-1}, y_j, \vec{x}, j\right)\right) \tag{2}$$

The model parameters λ_i are set to maximize the penalized log-likelihood L on some training data T [20].

$$L(T) = \sum_{(\vec{x},\vec{y})\in T} \log p\left(\vec{y}\,|\,\vec{x}\right) - \sum_{i=1}^{m} \frac{\lambda_i^2}{2\sigma^2} \tag{3}$$

The partial derivations of $L(T)$ are:

$$\frac{\partial L(T)}{\partial \lambda_i} = \tilde{E} f_i - E(f_i) - \frac{\lambda_i}{\sigma^2} \tag{4}$$

Where $\tilde{E}(f_i)$ is the empirical expectation of feature f_i and can be calculated by counting the occurrences of f_i in T. $E(f_i)$ is the model expectation of f_i and can be written as:

$$E(f_i) = \sum_{(\vec{x},\vec{y})\in T} \sum_{\vec{y}'\in Y^n} P_{\vec{\lambda}}\left(\vec{y}\,|\,\vec{x}\right). \sum_{j=1}^{n} f_i(y'_{j-1}, y'_j, \vec{x}, j) \tag{5}$$

Direct computation of $E(f_i)$ is intractable due to the sum over all possible label sequences $\vec{y}' \in Y^n$. The Forward-Backward algorithm [25] solves this problem efficiently. Forward (α) and backward (β) scores are defined by

$$\alpha_j\left(y\,|\,\vec{x}\right) = \sum_{y'\in T_j^{-1}(y)} \alpha_{j-1}\left(y'\,|\,\vec{x}\right).\Psi_j(\vec{x}, y', y) \tag{6}$$

$$\beta_j\left(y\,|\,\vec{x}\right) = \sum_{y'\in T_j(y)} \beta_{j+1}\left(y'\,|\,\vec{x}\right).\Psi_j(\vec{x}, y, y') \tag{7}$$

Where $\Psi_j\left(\vec{x}, a, b\right) = \exp(\sum_{i=1}^{m} \lambda_i f_i\left(a, b, \vec{x}, j\right))$, $T_j(y)$ is the set of all successors of a state y at a specified position j, and, accordingly, $T_j^{-1}(y)$ is the set of predecessors.

Normalized forward and backward scores are inserted into Eq. (5) to replace $\sum_{\vec{y}' \in Y^n} P_{\vec{\lambda}}\left(\vec{y} \mid \vec{x}\right)$ so that $L(T)$ can be optimized with gradient-based or iterative-scaling methods.

The rest of this paper is organized as follows: Section 2 reviews the related works. Section 3 briefly introduces the learner model variance and the method to minimize the model's error rate based on it. Section 4 describes our proposed method in detail. Experimental results are reported in Sections 5, and we conclude in Section 6.

2. Related work

The first mix model of active and semi-supervised learning was applied to assign class labels to unlabeled examples and introduced by McCallum and Nigam [2]. They combined the active learning algorithm with EM style semi-supervised learning to assign class labels to those examples that remain unlabeled. Similarly, Muslea et al. introduced a new multi-view algorithm, Co-EMT, which combines semi-supervised and active learning for text classification. Exploiting multiple views for both active and semi-supervised learning has been shown to be very effective [15]. Contrary to our approach, both of these two works applied their proposed combined approaches for text classification and they had a document-level view of data.

Yao et al. proposed a combination of active learning and self-training method to reduce the labeling effort for Chinese Named Entity Recognition. A new strategy based on Information Density (ID) for sample selecting in sequential labeling problem is also proposed in this work, which is claimed that is suitable for both active learning and self-training. CRF is used as the underlying model for active learning and self-training in their approach [26].

Xu et al. proposed a semi-supervised semantic annotation method for Chinese language. The method is called self-teaching SVM-struct and it uses fewer labeled examples to improve the annotating performance. The key of their self-teaching method is how to identify the reliably predicted examples for retraining. Two novel confidence measures are developed to estimate prediction confidence [23]. Although there is a similarity between the goal of this work and the one in ours, differences in Chinese language had them to consider the data in character-level and consequently, they modified the self-teaching method in that manner.

Another work closely related to ours is that of Tomanek and Hahn [20]. They propose an approach to AL where human annotators are required to label only uncertain subsequences within the selected sentences, while the remaining subsequences are labeled automatically based on the model available from the previous AL iteration round. They use marginal and conditional probability as confidence measure estimator and they only apply the semi-supervised scenario as an auxiliary part beside the AL.

In contrast, we use the model's variance as a new confidence measure estimator and also our approach has a distinct self-training phase in addition to a combined semi-supervised active learning phase. Furthermore, while we use segments of data instead of considering the data set as a whole in each iterations, the calculation of finding confident examples would be faster and the self-training phase produce more flexible and reliable result regards to primitive chunks and weakness of the current model.

In [1], they propose a range of active learning strategies for IE that are based on ranking individual sentences, and experimentally compare them on a standard dataset for named entity extraction. They have argued that, in active learning for information extraction, the sentence should be the unit of ranking.

Cheng et al. propose an active learning technique to select the most informative subset of unlabeled sequences for annotation by choosing sequences that have largest uncertainty in their prediction. Their active learning technique uses dynamic programming to identify the best subset of sequences to be annotated, taking into account both the uncertainty and labeling effort [11]. In contrast to our work, they only exploited AL scenario for sequence labeling task and also used SVM$^{\mathrm{Struct}}$ as the base classifier in their approach.

A method called BootMark, for bootstrapping the creation of named entity annotated corpora presents in [10]. The method requires a human annotator to manually mark-up fewer documents in order to produce a named entity recognizer with a given performance, than would be needed if the documents forming the base for the recognizer were randomly drawn from the same corpus.

3. Variance reduction for minimizing learner model error

Recently, several surveys have been done on application of different Active Learning methods in Named Entity Recognition and other NLP tasks [5,19]. By reviewing these surveys and other related works, it's revealed that variance based error reduction has been never used in active learning approaches that have been proposed in this field. In this section, we briefly review the model or classifier variance in the case that, firstly each instance x is added to L_n (initial labeled set), and then we argue that minimizing the classifier variance is equal to minimizing its error rate.

Let x be an input instance, let c_i be a class label and let $p(c_i|x)$ be a classifier's probability estimation in classifying x, then the actual probability $f_{c_i}(x)$ is shown as follow:

$$f_{c_i}(x) = p\,(c_i \,|\, x) + \varepsilon_i(x) \tag{8}$$

Where $\varepsilon_i(x)$, is an added error. If we consider that the added error of the classifier mainly comes from two sources, i.e., classifier bias and variance [16], then the added error $\varepsilon_i(x)$ in Eq. (8) can be decomposed into two terms, i.e., β_{c_i} and $\eta_{c_i}(x)$, where β_{c_i} represents the bias of the current learning algorithm, and $\eta_{c_i}(x)$ is a random variable that accounts for the variance of the classifier (with respect to class c_i), which gives [28]:

$$f_{c_i}(x) = p\,(c_i \,|\, x) + \beta_{c_i} + \eta_{c_i}(x) \tag{9}$$

According to [21] classifiers that trained by using the same learning algorithm but different versions of the training data suffer from the same level of bias but different variance values. This fact is shown in Fig. 2.

Assuming that we are using the same learning algorithm in our analysis, without loss of generality, we can ignore the bias term. Consequently, the learner's probability in classifying x into class c_i becomes

$$f_{c_i}(x) = p\,(c_i \,|\, x) + \eta_{c_i}(x) \tag{10}$$

According to [22] it's concluded that, the classifier's expected added error can be defined as:

$$\mathrm{Err}_{\mathrm{add}} = \frac{\sigma^2_{\eta_c}}{S} \tag{11}$$

Where $\sigma^2_{\eta_c}$ denotes the variance. As it's indicated Eq. (11), the expected added error of a classifier is proportional to its variance; thus, reducing this quantity reduces the classifier's expected error rate.

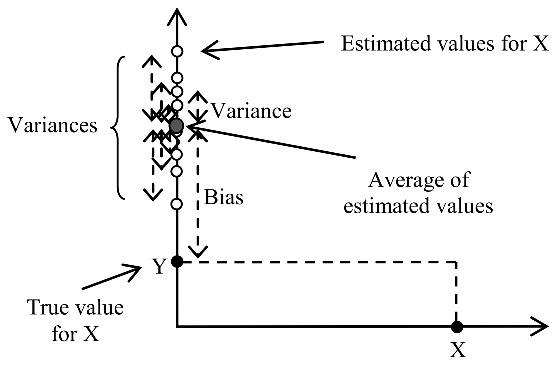

Fig. 2. Bias and variaces for a classifier.

In the same time, for a learner model which is trained on a training set L, the classifier variance for instance x is calculated by:

$$\sigma^2_{\eta^\theta_{c_i}} = \frac{1}{|\Upsilon_x|} \sum_{(x,c)\in\Upsilon_x} \left(y^x_{c_i} - f^\theta_{c_i}(x)\right)^2 \tag{12}$$

Where θ is the current learner model and Υ_x is a temporary set that its elements are defined by:

$$\Upsilon_x = L_n \cup < x, \bar{y} : \theta > \tag{13}$$

While $\bar{y}$ is the estimated target label for x which is predicted by θ. In Eq. (12), $|\Upsilon_x|$ denotes the number of examples in Υ_x. Through this analysis, our presented confidence measure base on variance reduction is:

$$\varphi = \sum_{i=1}^{l} p\,(c_i \,|\, x;\theta)\sigma^2_{\eta^\theta_{c_i}} \tag{14}$$

In the rest of the paper we use V_{MEASURE} instead of φ symbol for indicating our variance based confidence measure. To find an informative token x in a given sequence, at first the value of Eq. (14) (i.e. V_{MEASURE}) is calculated for each token along with their predicted labels. Then by comparing these values and through a decision scenario which would be described in Section 4, the informative tokens would be selected.

4. Two phase hybrid framework of semi-supervised and active learning

Semi-supervised learning and active learning can be combined to construct stronger active learners using unlabeled examples. We proposed a hybrid framework of semi-supervised learning and active learning at two levels and two distinct phases. This hybrid framework depends on the nature of the data in sequence labeling problem, hence we explain the ways that these textual data can be viewed.

Generally there are three meaningful views for textual data (specifically in English language); document-level view, sequence-level view, and token-level view. In document classification problems, textual data are viewed in document-level, and in sequence labeling problems, there is a sequence-level

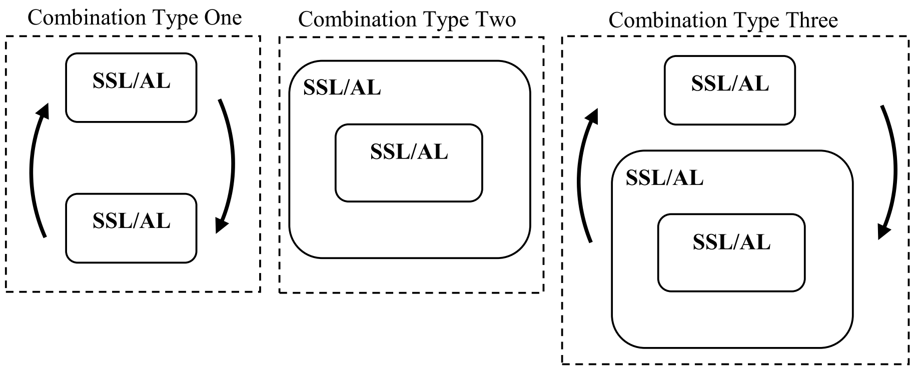

Fig. 3. Three types of hybrid framework for combining SSL and AL.

view, and in problems such as NER, the textual data are usually viewed in token-level (and in some approaches, both in token-level and sequence-level view).

By means of the above definition, we classify the ways that semi-supervised learning and active learning can jointly perform in three groups. In the first type of this combination (which we named it combination Type One) there are two continuous phases that in each of them a semi-supervised learning or active learning method can be conducted. In fact, the learner model is trained in each phase and then is handed out to the other phase for further learning. In this type of hybrid framework there is only single-level view of data set, like the typical structured data.

The second type of combining SSL and AL approaches has two-level view of data. In the combination Type Two, there is only one phase in which SSL and AL approaches corporate together. By having a multi-level view, this hybrid framework firstly considers data in sequence-level and performs a learning strategy (e.g. active learning) and would find the informative instances. After that, the other learning strategy (e.g. semi-supervised learning) is exploited in token-level to select the most confident tokens among those selected sequences.

Finally, the third way of joining SSL and AL (combination Type Three) has a two-level view on data and in fact it's a two phased framework of integration of Type One and Type Two. The first phase of this framework is a sequence-level SSL or AL method and the second phase is the same as the combination Type Two. The first and second type of combination of SSL and AL approaches have been used in some proposed hybrid methods, but due to our knowledge, the third type is a new hybrid framework for joining these two learning approaches.

We've applied combination Type Three in our approach and for this reason we named our framework 2L-DP SSAL (2-Level Double Phased Semi-supervised and Active Learning hybrid framework). The first phase, in this framework, is a sequence-level self-training and the second phase is a token-level combination of variance based active learning and self-training which is applied the confidence measure that presented in Section 3. The over all process of our framework is shown in Fig. 4.

We have also proposed 1L-SP SSAL approach which is a one phased combination of AL and SSL [12]. The overall process of that approach is similar to the second phase of 2L-DP SSAL. So, we don't define 1L-DP SSAL in detail in this paper, but we use its results for comparison.

While most text data sets are huge, applying SSL and AL approaches on whole of them may require a lot of time to reach a desirable result. To face this obstacle, we split the data set into segments and consider each segment as a unit that learning process must be applied for it. By dividing the unlabeled data into segments, the leaning scenarios can be performed separately on each of them. In self-training

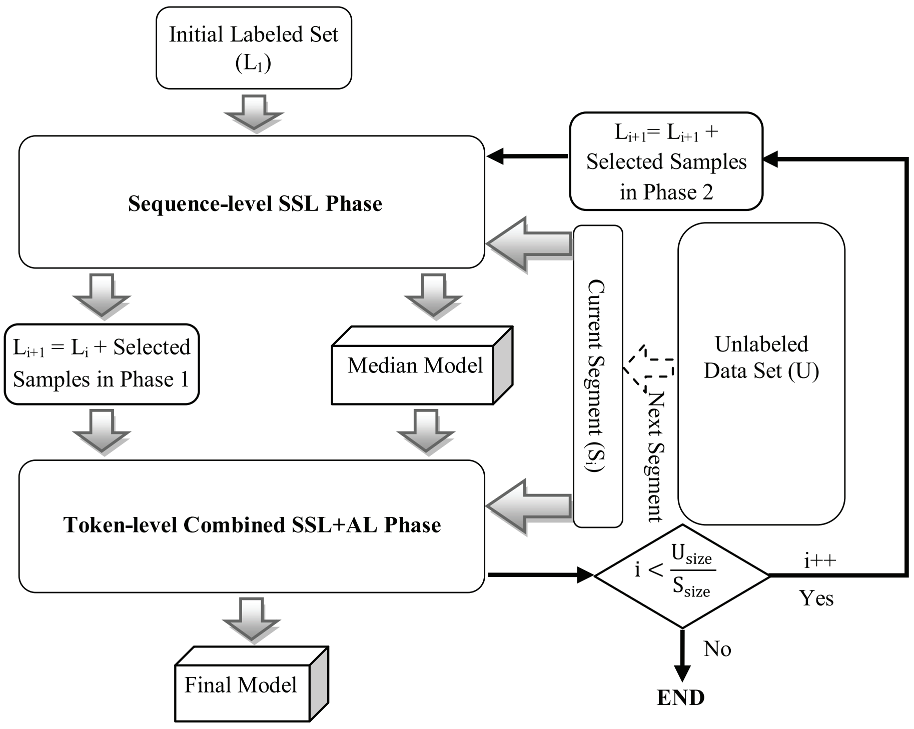
Initial Labeled Set (L_1)
Sequence-level SSL Phase
L_{i+1}= L_{i+1} + Selected Samples in Phase 2
L_{i+1} = L_i + Selected Samples in Phase 1
Median Model
Current Segment (S_i)
Next Segment
Unlabeled Data Set (U)
Token-level Combined SSL+AL Phase
$i < \frac{U_{size}}{S_{size}}$
i++
Yes
No
END
Final Model

Fig. 4. The framwork of 2L-DP SSAL.

algorithm the performance of the model significantly depend on the initial labeled set. Usually the size of the initial labeled set is too small for producing an accurate model, so it's better to rely less on the initial model. The first benefit of segmentation is that, by boosting the model in the second phase of the framework, the performance of the self-training algorithm would improve dramatically for next segments. This segmentation would enhance the overall performance and it would reduce the time needed for processing learning scenarios. The process of 2L-DP SSAL method is shown in Algorithm 3.

In Fig. 4, "i" indicates segment number, U_{size} is the number of samples in unlabeled data set (U) and S_{size} is the number of samples in a segment. The learner model which is produced from the first phase is called Median Model. In the next subsections, we'll describe two phases of this framework in more details.

4.1. Sequence-level semi-supervised learning

Semi-supervised learning has been widely applied in different domains in which automation is a matter of concern. There are different semi-supervised learning methods such as self-training, co-training, graph-based methods, etc., however, self-training is one of the most applied and promising method among them. In our framework we use self-training method as the distinct semi-supervised phase, that by following it the most confidence sequences are picked up at beginning of the processing of each segment, and these sequences along with their predicted labels are then added to the labeled set.

We use CRF as the base learner in our hybrid framework. As described before, Conditional Random Fields (CRFs) are a probabilistic framework for labeling and segmenting sequential data. In the self-training process in 2L-DP SSAL, we use the conditional probability as a confidence measure for

Algorithm 3 Two phase SSL-AL

```
Input: Labeled Set $(L)$, Unlabeled Set $(U)$, Parameters for defining the number of instances which should be
labeled in each phase $(\alpha, \beta)$ and segment size $(S_{\mathrm{size}})$.
Output: Model $\theta$

Algorithm:
  1. Segmenting unlabeled data set $U$ and preparing current segment $(S_i)$,
  2. Loop for all Segments
    3. While reaching stop point (SSL$_{\mathrm{size}}$)  // Sequence level SSL phase continue until reaching the stop point
      4. Train CRF model on $L$,
      5. Find $K$ most confident sequences by means of their conditional probability,
      6. IF there is no confident sequence in $S_i$
        ○ End of SSL Phase
      7. ELSE
        ○ Add selected sequences with their predicted label sequences to $L$
    END While
    8. While reaching stop point (SSAL$_{\mathrm{size}}$) Token-level SSL-AL combination phase
      9. Train the model on current $L$,
      10. Find $K'$ least confident sequences by means of current model prediction probability.
      11. FOR each selected Sequences
        12. Calculate V$_{\mathrm{MEASURE}}$ for all tokens in the current Sequence,
        13. Calculate unit of decision, $unit = \frac{\max V_{\mathrm{MEASURE}} - \min V_{\mathrm{MEASURE}}}{\#\textit{of tokens in current Sequence}}$,
        14. Find minimum and maximum value among these tokens' variances,
        15. Find Least Confident (LCT) and Most Confident (MST) tokens based on their V$_{\mathrm{MEASURE}}$ and by
        means of unit of decision,
        16. Ask the label of LCTs from the Oracle,
        17. Automatically label MCTs in the sequences by means of the current model,
        18. Add the current sequence with its prepared label sequence to $L$,
        19. Train the model on current $L$,
      END FOR
      20. IF the condition $i < \frac{U_{\mathrm{size}}}{S_{\mathrm{size}}}$ is held
        ○ Prepare Next Segment $(S_{i+1})$,
        ○ Go to Step 2.
      21. ELSE
        ○ END of Phase 2,
    END While,
  END Loop,
END of Algorithm.
```

selecting reliable examples which is predicted by a trained CRF model for each sequences in unlabeled data set.

As it shown in Algorithm 3, the algorithm is start with segmenting the whole unlabeled data set (U). After segmentation, the self-training scenario is applied for the first segment. At the beginning of the self-training loop, the CRF model trains on the current labeled set (L). In the next step, a number of most confident sequences are selected from the current segment. These confident sequences are selected based on two conditions:

1. The estimated probability for the sequence must be greater than 0.99,
2. The sequence must consist of more than 2 tokens.

The first condition guaranties that throughout the self-training phase, only those sequences are automatically labeled that the current model is almost certain about them. By following the second condition, the algorithm would ignore the sequences that are too small. The small sequences rarely have enough information to train the learner model.

In each iteration of self-training, K number of most confident sequences (MCSs) are selected. The value of K parameter is defined as follow:

$$K = (\gamma \times SSL_{size}) \tag{15}$$

The γ is the parameter which defined the number of self-training iteration and the SSL_{size} is the number of sequences which are selected in SSL phase of our approach and is portion of segments size:

$$SSL_{size} = \alpha \times Segment_{size} \tag{16}$$

Here, the parameter α is a predefined portion that determines the maximum number of samples that can be annotated in phase 1 (SSL_{size}). The value of α is defined before running the algorithm.

Although the number of tokens which are annotated in phase 1 is restricted by SSL_{size}, after initial segments and when the model gains reasonable strength, this restriction can reduce the overall performance of the algorithm. To avoid this, by setting a condition at the end of phase 1 we examine that if all the determined numbers of samples are annotated in the current segment, a bonus value is added to the current SSL_{size} and this enhanced SSL_{size} is dedicated to the next segment.

$$SSL_{size} = SSL_{size} + Bonus_{size} \tag{17}$$

Through this discipline, we would compensate the model when it's able to find and predict all SSL tokens in a segment. This condition make our self-training algorithm behaves in a dynamic manner and avoids deterioration of performance when the model is in its infancy.

4.2. *Token-level hybrid of SSL and AL*

In second phase, we present a combined SSL and AL approach for annotating named entities. Like the general AL approach, at the beginning of phase two in Fig. 4, a predicting model is trained on a label set, that in our approach this training set includes initial label set in addition to the high confident examples which are labeled in phase one. After finding a number of least confidence sequences (LCSs) in unlabeled data set (based on the estimated probability by the current model), V_{MEASURE} values must be calculated for all tokens in each of these sequences (Eq. (14)). For separating most and least confident tokens (respectively MCTs and LCTs) by means of their V_{MEASURE}, a metric unit is calculated for making this decision. This unit is calculated as follow:

$$Unit = \frac{\max V_{\text{MEASURE}} - \min V_{\text{MEASURE}}}{Number\ of\ Tokens\ in\ current\ Sequence} \tag{18}$$

The way that we select informative tokens by means of the unit of decision is shown in Fig. 5.

For selecting the informative tokens, the distance between the minimum and maximum values of V_{MEASURE} which are calculated tokens in current sequence, is divided into units. Then, the informative tokens are selected based on the distribution of their V_{MEASURE} over this area. Those tokens which their V_{MEASURE} are between minimum V_{MEASURE} and minimum V_{MEASURE} plus unit, are selected as most confident tokens. It means that these tokens and their predicted labels make less changes or variation when they are added to the current labeled set, so their labels can be automatically predicted by the current model. Other tokens are treated as informative tokens which their labels must be asked from the Oracle. The number of samples that must be annotated in phase 2 is defined by:

$$SSAL_{size} = \beta \times Segment_{size} \tag{19}$$

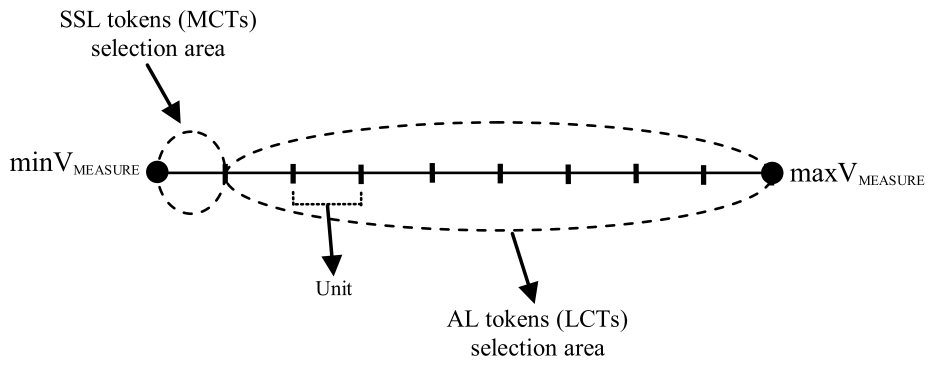
SSL tokens (MCTs) selection area
minV MEASURE
maxV MEASURE
Unit
AL tokens (LCTs) selection area

Fig. 5. Selecting informative tokens based on V_{MEASURE}.

The parameter β is a predefined ratio that determines the number of samples that selected in phase 2. To avoid reduction of performance by the effect of this restriction, after initial segments, the accuracy of the model is monitored and when it reaches to the convergence point, the value of β (and respectively $SSAL_{size}$) would be set to its maximum feasible number. The convergence of the accuracy of the model is assessed by this condition:

$$\left(\mathrm{F1}_{\mathrm{Segment[i]}} - \mathrm{F1}_{\mathrm{Segment[i-1]}}\right) < (\mathrm{F1}_{\mathrm{Segment[i-1]}} - \mathrm{F1}_{\mathrm{Segment[i-2]}})/2 \tag{20}$$

F1 or F-score is a well-known performance measure in information extraction domain that would be described in Section 5. A graphical illustration of the condition in Eq. (20) is shown in Fig. 6.

As shown in Fig. 6, we can say that the condition Eq. (20) is met when the distance “B” is less than the half of distance “A”. The advantage of the token-level combined framework in a NER task is that, even when a sequence is selected as an informative instance base on its low confident, it can still exhibit sub-sequences which do not add much to the overall utility and thus are fairly easy for the current model to label correctly. As it’s shown in Fig. 4, only those tokens within the selected sequences remain to be manually labeled, which have high variances due to the Eq. (14).

5. Experiments and results

5.1. Evaluation measures

We use two class of evaluation measures; performance measures and annotation cost. For evaluating the performance, the three metrics that widely used in the information retrieval field, precision, recall, and F-score, were adopted in this experiment. The precision and recall are calculated as follow:

$$Precsion = \frac{\#\ of\ correct\ answers\ found\ by\ the\ system}{\#\ of\ answers\ given\ by\ the\ system} \tag{21}$$

$$Recall = \frac{\#\ of\ correct\ answers\ found\ by\ the\ system}{\#\ of\ correct\ answers\ in\ the\ test\ corpus} \tag{22}$$

Actually, F-score is the harmonic average of precision and recall and is calculated as follow:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{23}$$

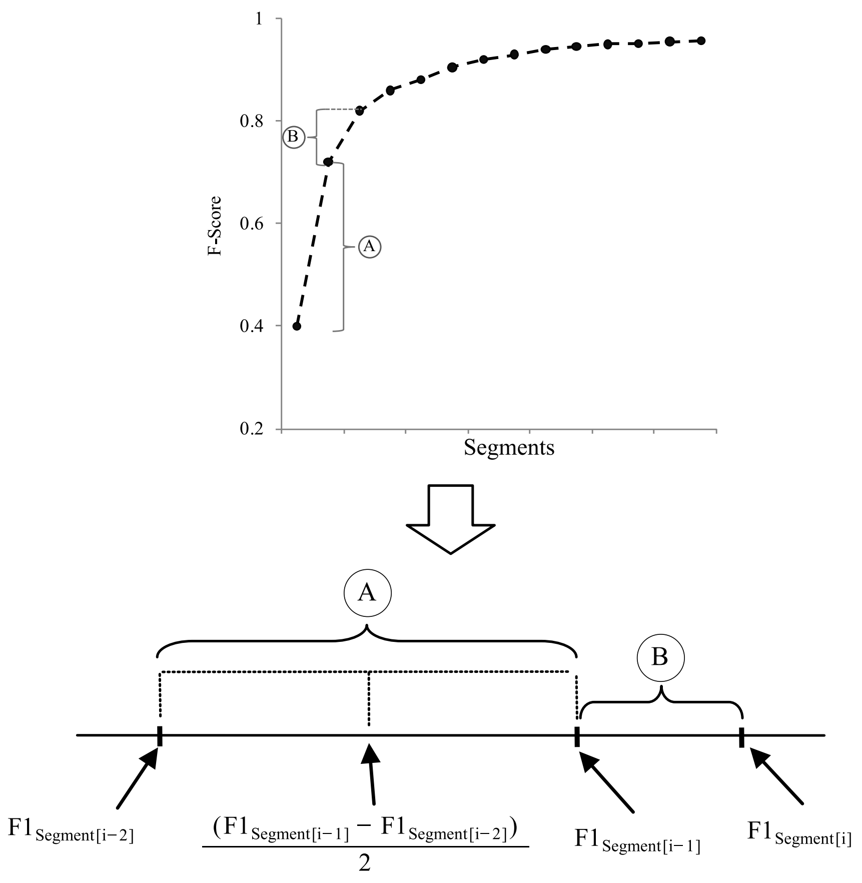

Fig. 6. Monitoring the convergence of learning curve.

For evaluating the cost of manually annotation, we've applied the Cost for Target Performance (CFP) measure which is introduced in [18]. For calculating the CFP, at first we must consider sampling complexity. Sampling complexity describes the number of examples needed to yield a particular target performance $perf(\theta_L, T)$ on the held-out test set T where θ_L specifies the model induced from the actively obtained sample L. The Cost for Target Performance (CFP) measure is proposed as an operationalization of sampling complexity. CFP quantifies the cost, according to an arbitrary cost measure, needed to obtain a target performance F^* given a sampling strategy S:

$$CFP_{F^*}(S) = \underset{cost(L)}{\arg\min}\, perf(\theta_L, T) \geqslant F^* \tag{24}$$

5.2. Experimental settings

The data set used in our experiment is CoNLL03 English corpus which is a well-known benchmark for Named Entity Recognition task [9]. The CoNLL03 corpus contains 4 label types which distinguish person, organization, location, and names of miscellaneous entities that do not belong to the previous three groups.

MUC7 corpus is one of the other benchmark data sets for NER task [30]. MUC7 corpus incorporates seven different entity types, viz. persons, organizations, locations, times, dates, monetary expressions, and percentages. The details of these two data sets are shown in Table 1.

Table 1
Characteristic of CoNLL03 English data

Sets	Entity classes	Sentences	Tokens
CoNLL-03 training set	4	14987	204567
CoNLL-03 test set	4	3684	46435
MUC7	7	3366	78305

For MUC7 we apply 10-fold cross-validation. On the CoNLL03 corpus, no cross-validation was performed because there is a designated training and evaluation set.

Both corpora are converted into IOB format, that is the label of the first token of an entity starts with B-EntityType and the rest of its tokens' labels (if it has more than 1 token) start with I-EntityType. The tokens outside the predefined types of entity are labeled as "O". The CoNLL03 basically is in IOB format, so we convert MUC-7 corpus into IOB format.

In our experiment we used the "MALLET" package as a CRF implementation [3]. MALLET is a collection of tools in Java for statistical NLP. An implementation of CRF in MALLET is used in our work. We employ the linear-chain CRF model in our system. All methods and classes are implemented in Java. A set of common feature functions was employed, including orthographical (regular expression patterns), lexical and morphological (suffixes/prefixes, lemmatized tokens), and contextual (features of neighboring tokens) ones. Unlike several previous studies, we did not employ additional information from external resources such as gazetteers. All our features can be automatically extracted from the supplied data.

We split the data set in segments with 500 sequences and separately applying our two phase framework for each segments. Overall experiment start from a 20 randomly selected sequences as initial label set (L). Our two phase SSL-AL method pick up 100 sequences in 2 iterations, and this process is applied for each segment.

5.3. Features set

We employ a rich set of standard token-level features for NER. These include the word itself, various orthographic features such as capitalization, the occurrence of special characters such as hyphens, suffixes and prefixes, and context information in terms of features of neighboring tokens to the left and right of the current token. A detailed description of features typically used for NER is introduced in [8]. These features are very suitable and general enough to be used in most (sub)domains for entity recognition. Moreover, there has been discussion that CRFs are able to handle such large amounts of presumably highly correlated features.

5.4. Experimental results

Many works on NER, especially in the biomedical domain, have shown that the performance of a CRF model can be immensely increased when the standard feature set is optimized and extended in an appropriate way [27]. However, throughout this work we employ the same standard feature set for comparability of all experiments. That's because of the fact that, the focus of this work is not on feature selection and outperforming state-of-the-art performance of NER when given huge amounts of training data, but on cost-efficient ways to provide highly useful training material. Our standard feature set does, though, yield respectable performance values as shown in the rest of this section. Note that, while the CoNLL03 corpus is prepared with NP chunks and POS tags, we didn't omit these features.

Table 2
Results of 2L-DP SSAL method on CoNLL03 (manually labeled tokens = 14618)

	Precision	Recall	F1-score
PER	0.8436	0.8442	0.8439
ORG	0.7787	0.7762	0.7775
LOC	0.8684	0.8649	0.8666
MISC	0.8467	0.7842	0.8242
OVERALL F1-score		0.8414	
Token accuracy		0.9785	

Table 3
Results of supervised CRF on CoNLL03 (on the whole of the train set)

	Precision	Recall	F1-score
PER	0.8638	0.8572	0.8605
ORG	0.7818	0.7889	0.784
LOC	0.8543	0.8841	0.8689
MISC	0.8505	0.7792	0.8223
OVERALL F1-score		0.8423	
Token accuracy		0.9763	

Table 4
Analysis of 2L-DP SSAL on CoNLL03

Annotated tokens				
Manual	Automatic	$\sum$	AR (%)	ACC
1000	99	1099	90.99	93.68
2000	3641	5641	35.45	94.36
3000	5853	13653	27.29	95.08
4000	14191	18791	21.28	95.35
10000	53623	63623	15.71	96.89
14618	97556	112174	**13.03**	97.58

Table 5
Values of $CFP_{0.79}$ on CoNLL03 and $CFP_{0.85}$ on MUC-7

	2L-DP SSAL	TM	1L-DP SSAL	FuSAL	RS
CoNLL03	9014	12117	14210	36014	49983
MUC-7	5697	11001	12015	18516	37518

One of the recent works for sequence labeling problem which is closely related to ours is [20], hence we compare the results of our approach to theirs in a similar setting. In the rest of the paper we refer to their work by TM. In addition, to show the impact of semi-supervised learning in our approach, we present the results of a sequence-level fully supervised active learning (FuSAL) with CRF as its base learner model.

Tables 2 and 3 respectively show the result of 2L-DP SSAL approach and a completely supervised learning approach (with CRF) on CoNLL03 corpus. These results are produced by training the learner model on CoNLL03 train set and evaluate the model on its test set.

As it's depicted in Tables 2 and 3, for CoNLL03 corpus, the 2L-DP SSAL approach showed the same accuracy as completely supervised CRF approach in the case that it only required 14618 manually labeled tokens which is about only 8% of the number of all tokens in the train set. More information about the portion of the number of manually labeled tokens to automatically labeled tokens which were used in 2L-DP SSAL is shown in Table 4.

The annotation rate (AR) in Table 4 is the portion of manually labeled tokens on the total amount of labeled tokens. It shows that the annotation rate in 2L-DP SSAL meanwhile its f-score is equal to supervised manner is only 13.03%.

For comparing 2L-DP SSAL with other approaches base on annotation cost we use CFP measure. For the CFP scores, a corpus-specific target performance F* is chosen so as to be as large as possible, with the constraints that mentioned in [18]; (a) it should occur before the convergence phase and (b) so that all metrics reach this score within a maximum of 50,000 tokens. Random sampling (RS) is taken as a baseline scenario for this comparison. Table 5 shows the results for different approaches for F* = 0.79 ($CFP_{0.79}$) on CoNLL03 corpus and F* = 0.85 ($CFP_{0.85}$) on MUC-7 corpus.

The learning curve of these approaches based on the number of manual labeled tokens is shown in Figs 7(a) and (b) respectively on CoNLL03 and Muc-7 corpora.

As Figs 7(a) and (b) reveal, the learning curves of 2L-DP SSAL stopped early (on MUC7 after 7108 tokens and on CoNLL03 after 26148 tokens), because at that point the whole corpus has been labeled

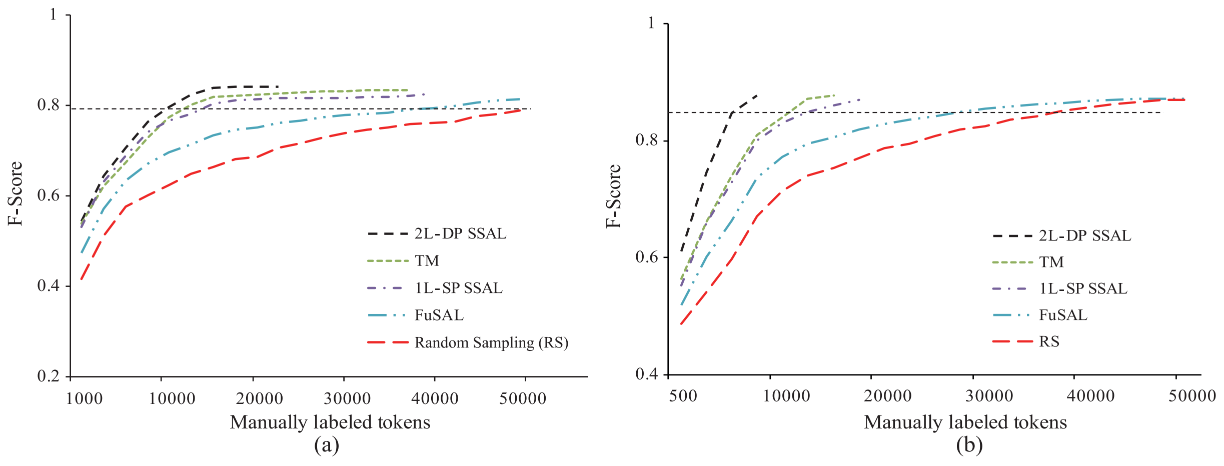

Fig. 7. Learning curves for 5 approaches (a) on CoNLL03, (b) on MUC-7. (Colours are visible in the online version of the article; http://dx.doi.org/10.3233/IDA-130577)

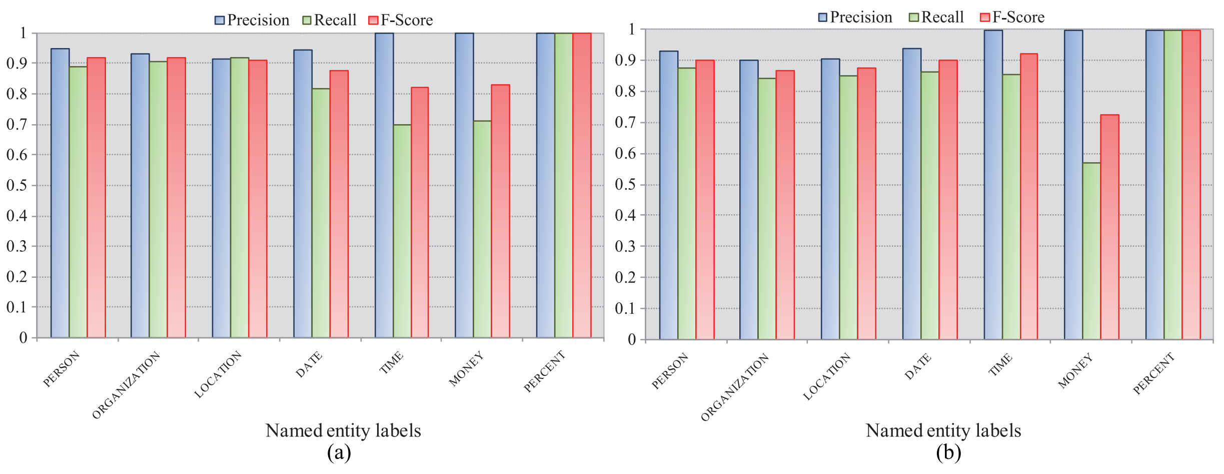

Fig. 8. Performance evaluation on MUC7, (a) 2L-DP SSAL approach (Manually labeled tokens = 6056), (b) Supervised CRF (On the whole of the data set). (Colours are visible in the online version of the article; http://dx.doi.org/10.3233/IDA-130577)

exhaustively – either manually, or automatically. So, by using 2L-DP SSAL the complete corpus can be labeled while only a small fraction of its data is manually annotated (MUC7: about 9%, CoNLL03: about 13%) and the rest is automatically annotated.

The results of 2L-DP SSAL approach and supervised learning on MUC7 corpus is shown in Fig. 8. Our approach reached the accuracy of a completely supervised method when it only needed 6056 manually labeled tokens, i.e. only 8% percent of the number of all tokens in the data set.

The statistics of the proportion of the number of manually labeled tokens to automatically labeled tokens of 2L-DP SSAL approach on MUC7 corpus is shown in Table 6. It's revealed that at the end of the algorithm, the Annotation Rate was only 12.12 percent which means almost 90% of annotation task was done automatically.

To compare with the AR values for CoNLL03, in MUC-7 the AR value didn't begin from a high percent (such as 90.99%) like CoNLL03, but it starts from a considerably lower percent (48.7%). One

Table 6
Analysis of 2L-DP SSAL on MUC7

Annotated tokens				
Manual	Automatic	$\sum$	AR (%)	ACC
1000	1053	2053	48.7	95.24
2000	4554	6554	30.51	96.28
3000	8420	11420	26.26	97.02
4000	18172	22172	18.04	97.23
5000	26535	31535	15.85	97.49
6056	43871	49927	**12.12**	97.72

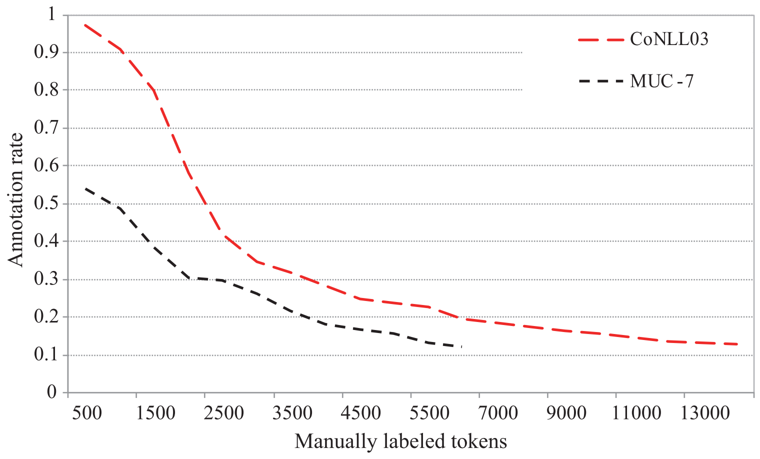

Fig. 9. Annotation Rate reduction in 2L-DP SSAL. (Colours are visible in the online version of the article; http://dx.doi.org/10.3233/IDA-130577)

of the reasons of this fact is the impact of the average length of the sentences and the distribution of the entities per sentences in these corpora. The average sentence length in CoNLL03 is 13 while in MUC-7 is 23 and the average number of entity occurs per sentence for CoNLL03 is 1.48 and for MUC-7 is 1.70. While the average sentence length in MUC-7 is greater than CoNLL03, but the distribution of entities over sentences is almost the same, it can be concluded that, most tokens in a sentence do not belong to entity classes, so their labels can be easily predicted by the model. In this condition, even the initial model can produce high portion of automatically annotated tokens, so the AR percent in the beginning of the algorithm on MUC-7 is not that high as in CoNLL03. The reduction of AR values by incrementing the segment number (from the beginning to the end of the algorithm) in 2L-DP SSAL approach is shown in Fig. 9.

The number of automatically labeled tokens in phase 1 on each segment is shown in Fig. 10 (for both CoNLL03 and MUC-7 corpora). Since the overall algorithm began from a very small initial labeled set, the primary model does not produce a reasonable accuracy, so in self-training phase of 2L-DP SSAL algorithm (phase 1), there is no confident tokens to be selected throughout the conditions in initial segment. It's a logical behaviour that must be expected from a self-training algorithm. As shown in Fig. 10 the number of automatically annotated tokens gradually increased from segment 2 to the last segment in both corpora.

Another characteristic of our combined method is that its final performance is not affected by the number of initial label set (L1). Unlike most of semi-supervised methods that their performance highly depend on the size of the initial labeled set, in our approach the initial labeled set has an extremely small size or in other words the proportion of initial labeled set size to the whole of the data set size is very small (almost between 0.15% to 0.55%). The reason of this behaviour is that in the active learning phase, the lack of samples in the initial labeled set would be compensated by the selected informative samples. The impact of initial L1 size on the cost of reaching to a specific target performance (here for F-score = 0.75) is shown in Fig. 11. The results of each of the different L1 size are generated from a 10-fold cross validation evaluation of 2L-DP SSAL on MUC-7 data set.

Figure 11 shows the number of tokens that should be annotated manually to reach the target performance in 2L-DP SSAL with changes in L1 size. The number of manually annotated tokens is shown in two cases to compare the real impact of growth of the L1 size. In case one, the number of tokens in L1

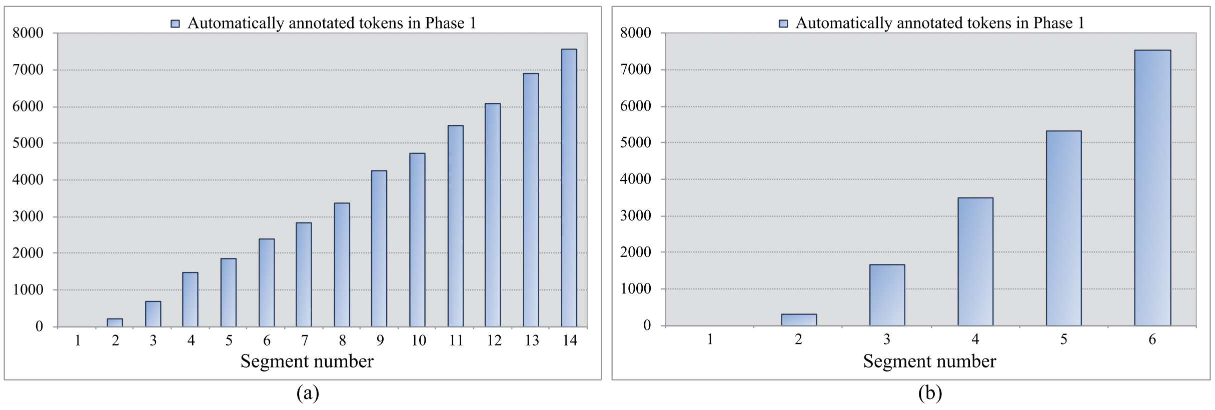

Fig. 10. Number of Automatically labeled tokens in phase 1 of 2L-DP SSAL approach, (a) CoNLL03 corpus, (b) MUC-7 corpus. (Colours are visible in the online version of the article; http://dx.doi.org/10.3233/IDA-130577)

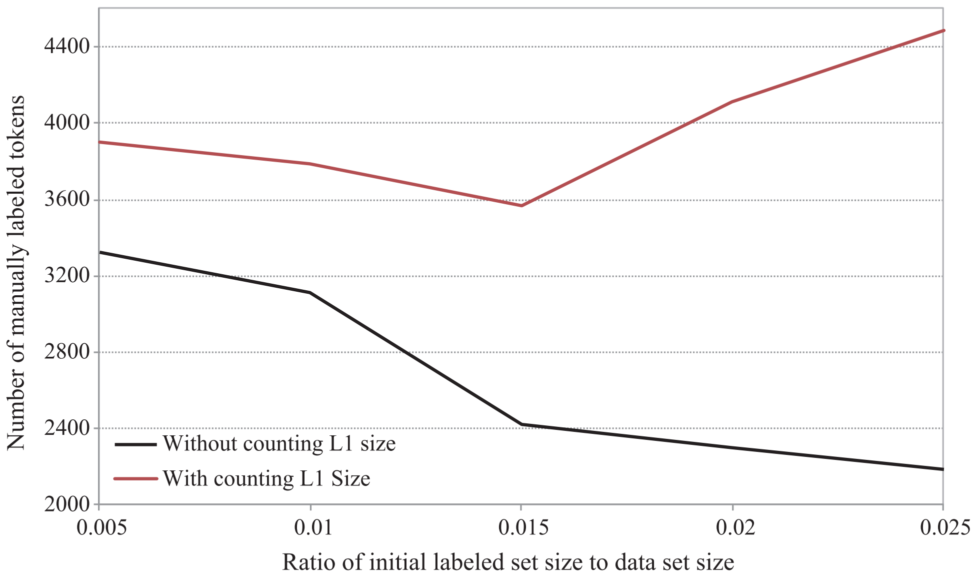

Fig. 11. Impact of the increment of L_1 size on manual labeling cost to reach a target performance (here F1 = 0.75 on MUC-7). (Colours are visible in the online version of the article; http://dx.doi.org/10.3233/IDA-130577)

set didn't augment with the manual labeling cost. In this case, the cost of reaching the target performance reduced steadily by the increase of L1 size. The sudden decrease of cost in the curve between 0.01 point and 0.015 is because of that, the initial model's accuracy would satisfy the two conditions in phase 1 of the 2L-DP SSAL even in the first segment, and some samples would be automatically annotated. In case two, the number of tokens in L1 set was considered as the manual labeling cost. The case two is more realistic than case one, because by increasing the size of L1 set the cost of manual labeling rose too. As it can be observed in Fig. 11, the cost of manual annotating slightly decreased at the beginning by the growth of L1 set size, but when the L1 size went up more, the cost increased considerably. This shows that the increase of L1 set size would not enhance the performance of our approach.

6. Conclusions

In this paper, we proposed a combination of active learning and semi-supervised learning by conducting a multi-level view to data and applied it for sequence labeling problem. Our hybrid framework consists of two phases, the first phase is a distinct self-training algorithm and the second phase is the hybrid SSL and AL approaches. In addition, we presented a confidence measure using learner model's variance, that to the best of our knowledge, it's the first attempt in sequence labeling domain. Conditional Random Fields was chosen as the underlying leaner model for the hybrid method. We used our new confidence measure for selecting the most informative tokens in the AL scenario.

Our experiments in the context of the NER scenario render evidence to the hypothesis that our two phased approach to semi-supervised and active learning for sequence labeling, indeed strongly reduced the amount of tokens to be manually annotated (in terms of numbers), about 90% compared to supervised learning and 60% to its fully supervised active learning counterpart. In comparison with one of the newest combined approaches for sequence labeling, our approach reduced annotation cost about 25% in terms of number of manually annotated tokens.

References

[1] A. Esuli, D. Marcheggiani and F. Sebastiani, Sentence-based active learning strategies for information extraction, in: *Proceedings of the 1st Italian Information Retrieval Workshop (IIR'10)*, Padova, Italy, 2010.

[2] A. McCallum and K. Nigam, Employing EM and pool-based active learning for text classification, in: *Proceedings of the International Conference on Machine Learning (ICML)*, Morgan Kaufmann, (1998), 350–358.

[3] A. McCallum, MALLET: A machine learning for language toolkit, in, 2002.

[4] B. Settles, Active learning literature survey, in: *University of Wisconsin-Madison*, 2009.

[5] B. Settles and M. Craven, An analysis of active learning strategies for sequence labeling tasks, in: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2008), 1069–1078.

[6] C. Campbell, N. Cristianini and A. Smola, Query learning with large margin classifiers, in: *Proceedings of ICML'00* (2000), 111–118.

[7] D.D. Lewis and J. Catlett, Heterogeneous uncertainty sampling for supervised learning, in: *Proceedings of the 11th International Conference on Machine Learning* (1994), 148–156.

[8] D. Nadeau and S. Sekine, A survey of named entity recognition and classification, *Linguisticae Investigation* **30** (2007), 2–26.

[9] E.F.T.K. Sang and F.D. Meulder, Introduction to the CoNLL- 2003 shared task: Language-independent named entity recognition, in: *Proceedings of CoNLL-2003*, Edmonton, Canada, (2003), 155–158.

[10] F. Olsson, On privacy preservation in text and document-based active learning for named entity recognition, in: *Proceeding of the ACM First International Workshop on Privacy and Anonymity for very Large Databases*, Hong Kong, China, 2009.

[11] H. Cheng, R. Zhang, Y. Peng, J. Mao and P.-N. Tan, Maximum margin active learning for sequence labeling with different length, in: *Proceedings of the 8th Industrial Conference on Advances in Data Mining: Medical Applications, E-Commerce, Marketing and Theoretical Aspects* (2008), 345–359.

[12] H. Hassanzadeh and M.R. Keyvanpour, A variance based active learning approach for named entity recognition, in: *International Conference on Intelligent Computing and Information Science, in Communications in Computer and Information Science (CCIS), Springer Berlin Heidelberg* **135** (2011), 347–352.

[13] H. Seung, M. Opper and H. Sompolinsky, Query by committee, in: *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory* (1992), 287–294.

[14] H. Wallach, Conditional random fields: An introduction, in, rapport technique MS-CIS-04-21, department of computer and information science, University of Pennsylvania, 2004.

[15] I. Muslea, S. Minton and C. Knoblock, Active+ semisupervised learning = robust multi-view learning, in: *Proceedings of International Conference on Machine Learning (ICML)*, Sydney, Australia, (2002), 435–442.

[16] J. Friedman, On bias, variance, 0/1-loss and the curse-ofdimensionality, *Data Mining Knowledge Discover* **1** (1996), 55–77.

[17] J. Lafferty, A. McCallum and F. Pereira, Conditional random fields: Probabilistic models for segmenting and labeling sequence data, in: *Proceeding of 18th International Conference on Machine Learning* (2001), 282–289.

[18] K. Tomanek, Resource-aware annotation through active learning, Doctor of Philosophy Thesis, Technical University of Dortmund, 2010.

[19] K. Tomanek and F. Olsson, A web survey on the use of active learning to support annotation of text data, in: *Proceedings of the NAACL HLT Workshop on Active Learning for Natural Language Processing*, Boulder, Colorado, (2009), 45–48.

[20] K. Tomanek and U. Hahn, Semi-supervised active learning for sequence labeling, in: *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP* (2009), 1039–1047.

[21] K. Tumer and J. Ghosh, Analysis of decision boundaries in linearly combined neural classifiers, *Pattern Recognition* **29** (1996), 341–348.

[22] K. Tumer and J. Ghosh, Error correlation and error reduction in ensemble classifier, *Connection Sci* **8** (1996), 385–404.

[23] K. Xu, S.S. Liao, R.Y.K. Lau, L. Liao and H. Tang, Self-teaching semantic annotation method for knowledge discovery from text, in: *42nd Hawaii International Conference on System Sciences*, 2009.

[24] Linguistic, Linguistic data consortium, *Message Understanding Conference* **7** (2001), LDC2001T02.

[25] L. Rabiner, A tutorial on hidden Markov models and selected applications inspeech recognition, *Proceedings of the IEEE* **77** (1989), 257–286.

[26] L. Yao, C. Sun, X. Wang and X. Wang, Combining self learning and active learning for chinese named entity recognition, *Journal of Software* **5** (2010), 530–537.

[27] R. Klinger, C. Kolarik, J. Fluck, M. Hofmann-Apitius and C. Friedrich, Detection of IUPAC and IUPAC-like chemical names, *Bioinformatics* **24** (2008), 268–276.

[28] R. Kohavi and D. Wolpert, Bias plus variance decomposition for zeroone loss function, in: *Proceedings of the 13th International Conference on Machine Learning* (1996), 275–283.

[29] X. Zhu and A.B. Goldberg, Introduction to semi-supervised learning, Morgan & Claypool, 2009.

[30] Y. Altun, I. Tsochantaridis and T. Hofmann, Hidden markov support vector machines, in: *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, Washington DC, (2003).

[31] Y. Freund, H. Seung and E. Tishby, Selective sampling using the query by committee algorithm, *Machine Learning* **28** (1997), 133–168.

[32] Y. Qi, P. Kuksa, R. Collobert, K. Sadamasa, K. Kavukcuoglu and J. Weston, Semi-supervised sequence labeling with self-learned features, in: *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining (ICDM09)* (2009), 428–437.